DE GRUYTER
OPEN
10.1515/tvsb-2016-0023

Transactions of the VŠB – Technical University of Ostrava
Civil Engineering Series, Vol. 16, No. 2, 2016
paper #23

Jan MAŠEK[1], Petr FRANTÍK[2], Miroslav VOŘECHOVSKÝ[3]

DESIGN OF EXPERIMENT USING SIMULATION OF A DISCRETE DYNAMICAL SYSTEM

Abstract

The topic of the presented paper is a promising approach to achieve optimal Design of Experiment (DoE), i.e. spreading of points within a design domain, using a simulation of a discrete dynamical system of interacting particles within an n-dimensional design space. The system of mutually repelling particles represents a physical analogy of the Audze-Eglājs (AE) optimization criterion and its periodical modification (PAE), respectively. The paper compares the performance of two approaches to implementation: a single-thread process using the JAVA language environment and a massively parallel solution employing the nVidia CUDA platform.

Keywords

Design of Experiment (DoE), design space, discrete dynamical system, pseudo-particles, massive parallelization.

1 STATE OF THE ART

Engineers must face the fact that many data needed for their decisions are uncertain or even inherently random. A decision must be made and one of the rational approaches is to consider a certain model of reality and consider selected inputs as random variables. Such a probabilistic approach is long accepted as a reasonable way to deal with uncertainties. As the model of a system under consideration (e.g. a finite element model of a structure) is often a complicated transformation of input variables, the full model is sometimes substituted by a simplified approximation – a surrogate model (response surface, neural network etc.). Such an approximation must be based and validated using an optimal set of input realizations. Statistical computations with the full model of the surrogate model must then be performed with realizations, i.e. values of the input variables weighted by the corresponding probability density functions.

The design and assessment of civil engineering structures using the currently valid standards recognize the existence of variability and uncertainty in design variables. The commonly used approach is the method of partial safety factors. This method yields acceptable results for typical structures and structural elements that are not too special and unique and repeat themselves in practice in limited variety. However, there are problems that require a specific, fully probabilistic approach and the standards allow for such a higher level of assessment. This can be the case of atypical or special structures due to their shape, acting loads or mass-produced elements (e.g., railway sleepers) which are worth of special attention. A definite reason for the use of fully probabilistic approach is a deployment of

[1] Ing. Jan Mašek, Department of Structural Mechanics, Faculty of Civil Engineering, Brno University of Technology, Veveří 331/95 602 00 Brno, Czech Republic, phone: (+420) 541 148 209, e-mail: masek.j@fce.vutbr.cz.

[2] Ing. Petr Frantík, Ph.D., Department of Structural Mechanics, Faculty of Civil Engineering, Brno University of Technology, Veveří 331/95 602 00 Brno, Czech Republic, phone: (+420) 541 147 376, e-mail: kitnarf@centrum.cz

[3] Prof. Ing. Miroslav Vořechovský, Ph.D., Department of Structural Mechanics, Faculty of Civil Engineering, Brno University of Technology, Veveří 331/95 602 00 Brno, Czech Republic, phone: (+420) 541 147 370, e-mail: vorechovsky.m@fce.vutbr.cz

a newly developed material or construction system for which a design approach and values of safety coefficients are not specified.

As mentioned above, the computational model of a structure can be viewed as a transformation of input random variables. These usually describe the dimensions of structural elements, material properties and the magnitude and duration of acting loads during the structural service life. Only in case of limited number of rather simplified models with low number of input random variables, it is possible to consider their mutual relationships and, using direct transformation, obtain probabilistic description of the output. This is, however, not the case of models of most real structures for which the analytical approach cannot be used due to the complexity of the models. One needs to estimate parameters of the output random vector using statistical analysis, i.e. by performing repeated simulations in the spirit of the Monte-Carlo method. The number of simulations will be denoted N_{sim}.

The most commonly used methods for estimation of statistical characteristics of structure response are the Monte-Carlo methods [1]: primarily the crude Monte-Carlo method (MC), Latin Hypercube Sampling (LHS) [2, 3], Importance Sampling (IS), and other. The estimation of statistical characteristics of response by a Monte-Carlo type method is performed as an approximation of integral using appropriately chosen integration points. The integration points are the selected sampling points the design of which is sometimes termed as the Design of Experiment (DoE).

Virtually all problems featuring various joint probability density of inputs can be transformed into selection of sampling points from a simple design domain: a unit hypercube [4]. This hypercube represents a space of independent uniformly distributed random variables. In fact, it represents the space of *sampling probabilities* of arbitrarily distributed variables. The dimension of the hypercube is N_{var}, which is the number of input random variables.

The execution of a sufficient number of simulations is usually a time-consuming task due to the complexity of the model. This reason leads to efforts aiming at minimization of the number of performed simulations N_{sim} while obtaining statistically significant characteristics of the output. Such a selection of optimal sampling set (DoE) is a part of every Monte-Carlo type method as the accuracy of the used method strongly depends on the choice of integration points.

In case of numerical integration using the Monte-Carlo method, the upper bound on the error of approximation can be estimated, for instance, by the Koksma-Hlawka inequality [5]:

$$\left| N^{-1} \sum_{i=1}^{N} f(x_i) - \int_0^1 f(x)\mathrm{d}x \right| \leq D(x_i)\, V(f), \tag{1}$$

where the left hand side of the inequality is the error of estimation using the average obtained by sampling analysis and $D(x_i)$ is the discrepancy, or the rate of uniformity of distribution of integration points, and $V(f)$ is the variation of integrated function. From the above mentioned it is clear that for achieving a robust design of integration points for a general, piecewise smooth function, the bound on error can be minimized only by obtaining as uniform distribution of integration points in design space as possible. The presented paper proposes a way of obtaining such design of integration points. The goal then is to distribute N_{sim} of design points within a N_{var}-dimensional hypercube as evenly as possible. Such a process can be understood as an optimization problem, the objective of which is minimization of a given functional.

The problem at hand (a uniform distribution of finite number of points within a design space) is relevant not only to structural design, but it is also featured in many other engineering tasks, operational research, computer modeling and also in real experimental research.

One of possible optimization methods is sampling by using the LHS method [6, 7], where the initial coordinates of N_{sim} design points are selected for each individual variable from N_{sim} equidistant subintervals of length of $1/N_{\text{sim}}$, each of which contains a single point, see figure 1a. By using the LHS method, a perfectly uniform distribution along each separate dimension is achieved. The entire volume of unitary hypercube can be so divided into $N_{\text{sim}}{}^{N_{\text{var}}}$ parts of equal volume. The remaining task is to minimize a chosen optimization criterion by mutual swapping of coordinates. It should be noted that

there exist $(N_{\mathrm{sim}}!)^{N_{\mathrm{var}}-1}$ different combinations of these coordinates (sampling plans). The time requirements for solving such an optimization problem therefore grows steeply with the number of simulations N_{sim} and with the number of dimensions N_{var} (this task is known to be an NP-hard problem [8]).

Another way of design of point layout is generating of initial coordinates of N_{sim} points using a pseudorandom number generator. Then, the distribution of points within the design space is optimized by one of possible optimization techniques, such as simulated annealing [8] or genetic algorithm [9, 10].

An optimal selection of the sampling points can also be performed using some of many criteria proposed in the literature to obtain uniformity of point distribution of low discrepancy of the design. For example, let us mention the criteria Audze-Eglajs (AE) [11], MaxiMin and MiniMax [12], Modified [13], Centered [14] or Wrap-Around [15] L2 discrepancy, Voronoi tessellation [16].

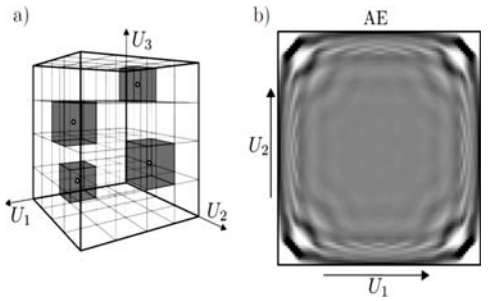

Fig. 1: a) Design hypercube ($N_{\mathrm{var}} = 3$) evenly divided into cells of equal volume. The design points are highlighted ($N_{\mathrm{sim}} = 4$). b) Visualization of a relative density of distribution of design points within a two-dimensional domain obtained by LHS optimized via AE criterion. The white color corresponds to zero density while the dark gray color corresponds to four times the average density (figure adapted from [19]).

It has been already shown [17, 4, 19] that the utilization of the AE criterion and also other criteria with intersite distances between points for the optimization of layout of design points in bordered space does not lead to a statistically uniform distribution. Due to the effect of boundaries of the hypercube volume, overly and also insufficiently filled regions tend to emerge. Using of such set of integration points for a Monte-Carlo type method leads to systematically biased results. A perfectly uniform distribution can be obtained considering the periodic extension of design hypercube [4, 19], see figure 2a. The mentioned modification of the original AE criterion (PAE) provides absolutely even distribution of points, see figure 1b.

Some of the mentioned criteria are based on a physical analogy between layout of integration points within the volume of the hypercube and a system of interacting mass points. Namely, the Audze-Eglājs criterion evaluates the amount of accumulated potential energy in a system of mutually repelling particles. The AE criterion assumes that the mutual repelling force between particles i and j is inversely proportional to third power of their mutual distance L_{ij}. In case of the PAE criterion [4,19], the shortest distance $\bar{L}_{ij}$ within the periodically expanded design space is considered, see figure 2a. The formula for evaluating the potential energy of the system E^{PAE} can be written as follows:

$$E^{\mathrm{PAE}} = \sum_{i=1}^{N_{\mathrm{sim}}} \sum_{j=i+1}^{N_{\mathrm{sim}}} \frac{1}{\bar{L}_{ij}^2}, \tag{2}$$

$$\text{where } \bar{L}_{ij} = \sqrt{\sum_{v=1}^{N_{var}} \left[\min\left(\Delta_{ij,v}\,; 1 - \Delta_{ij,v}\right)\right]^2}, \tag{3}$$

with $\Delta_{ij,v} = |x_{i,v} - x_{j,v}|$ being the projection of the distance of points *i* and *j* onto dimension *v*. Increasing the power in (2) to infinity corresponds to transition from AE criterion to MiniMax criterion (maximization of the distance between the closest pair of points).

2 ANALOGY BETWEEN DOE AND A DISCRETE DYNAMICAL SYSTEM

Instead of utilizing of the AE/PAE criterion as a norm for the combinatorial optimization of initial coordinates, one can consider solving the problem of dynamical system of charged particles directly, see figure 2b. The coordinates of the particles of the dynamical system, after reaching a static equilibrium (minimization of potential energy), may be directly used as coordinates of design points within the unit hypercube.

According to previous results [20], it can be expected that obtaining of coordinates of design points by using simulation of dynamical system will be significantly more efficient than the combinatorial and heuristic approaches [7]. The proposed approach will also enable solving the DoE issues mentioned below.

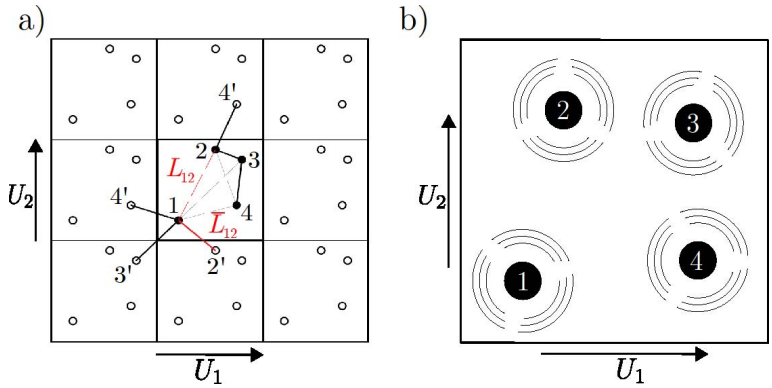

Fig. 2: a) Periodical extension of design space. b) System of repelling particles.

In practice, after executing a set of designed simulations, there often rises a need of executing several more simulations with coordinates of additional design points [18]. There is, of course, the possibility of designing the layout of the new set of design points N_{sim}. However, this would mean discarding of precious obtained results with design point from a current design. A rational approach therefore is the addition of new design points to the current design and optimization of their coordinates while preserving the coordinates of already performed simulations.

The above-mentioned requirements of design of experiments (periodical extension of design space and optimization of coordinates of newly added points while preserving the results of previously performed simulations) can be satisfied by the simulation of discrete dynamical system.

The aim of long-term efforts of the authors is to develop a robust tool for solving of a discrete dynamical system of interacting particles. By utilization of efficient numerical simulations, the influence of various constitutive laws and optimization criteria will be investigated. Special attention will be paid to the development of a criterion with the ability to consider the uniformity of point distribution not only within the volume of entire design space of dimension $N_{var,}$ but also within all subspaces of lower dimension.

3 COMPUTER IMPLEMENTATION

The initial implementation [20] of dynamical system simulation has been performed in the environment of JAVA programing language. The implementation, utilizing features of object-oriented programing, is executed by a single CPU thread. One can therefore expect that the time requirements of the solution will rise steeply with the number of points within the design hypercube N_{sim} and especially with the dimension of the hypercube N_{var}. The motivation for such kind of implementation was the possibility of rather transparent verification of system behavior and investigation of solution stability. Along with the implementation of dynamical system simulation, a graphic user interface enabling the control and observation of the simulation was created.

After a deeper investigation of the nature of the problem at hand, one can state that solving of such a task could advantageously implemented using massive parallelism. That means a concurrent solution of independent instructions on many threads operated by multiple cores of a graphic processor unit. Using a GPU for tasks other than image processing is known as GPGPU (**G**eneral-**P**urpose computing on **G**raphics **P**rocessing **U**nits). The two most exploited options are nVidia CUDA [21] and ATI FireStream [22] platforms. Such implementation is then performed in modifications of low-level programing languages, namely the C/C++ family or FORTRAN.

For the implementation of parallelized solution of the discussed dynamical system, the nVidia CUDA platform and the CUDA C/C++ programming language were chosen. The idea of massive parallelism is to divide the solution between many parallel (concurrently running) threads which typically execute identical independent instructions. In case of the solution of dynamical system simulation, one entire step of the dynamical simulation can be divided into the following substeps:

- calculation of components of distance vectors of all point pairs,
- calculation of absolute values of distance vectors of all point pairs,
- calculation of repelling forces between all point pairs (depends on selected constitutive law),
- numerical integration of equations of motion (semi-implicit Euler method).

Each of the listed substeps is executed by the highest possible number of concurrent threads performing identical tasks. The maximal number of concurrently running threads is, usually rather than by the hardware capabilities, limited by the nature of the implemented problem.

A fundamental issue of all parallelized implementations lies in the need to control the read and write events of each thread so that these do not access the same address in memory at the very same time [23]. This situation is known as the race condition and can occur in case of both reading and writing in the memory. When reading, threads which access the memory address later than other can read different value (changed by activity of preceding threads). When writing, on the contrary, the value at the observed memory address might be unintentionally overwritten by the rest of threads. The result of code exhibiting the race condition cannot be predicted as the time during which equally capable threads execute an identical instruction depends on non-deterministic circumstances.

Figure 3 offers a simplified illustration of the write-read procedure while performing the above mentioned parts of one step of dynamical simulation. The highest possible number of threads used for execution of each substep is limited by the size of the vector which is the subject of writing. In the figure 3, the solution of system of three points ($N_{sim} = 3$) in two-dimensional design space ($N_{var} = 2$) is shown.

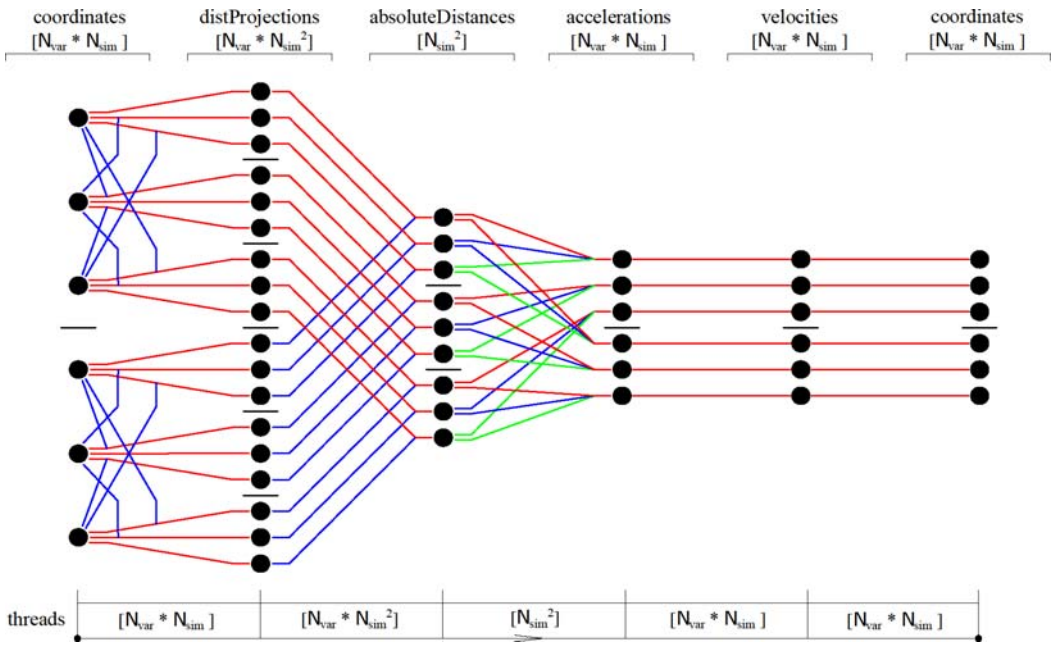

Fig. 3: Illustration of data structure and threads used for solving of individual parts of one step of dynamical simulation. Instructions requiring consecutive steps differ by color.

The first substep of solution is calculation of distance projections onto individual axes. These coordinate differences are calculated for all point pairs and for all dimensions and the output is stored in a vector of size $N_{var}*N_{sim}^2$; this can be executed without unnecessary delay using $N_{var}*N_{sim}^2$ active threads. Due to the sufficient hardware capabilities, there exist only two spots of performance drop (known as *bottlenecks*). Because of possible race condition while writing, it is necessary to add the squared projections along individual dimensions (to get the absolute distances) consecutively for each dimension. The same issue appears for increments of repelling forces between particles: it necessitates consecutive addition of forces from all N_{sim} points; that represents N_{sim} consecutive additions for each point along each single dimension. The following numerical integration using semi-implicit Euler method utilizes an ideal thread count. That means $N_{var}*N_{sim}$ threads.

After considering the above mentioned, one can expect the time requirements of parallelized solution to depend on both point count N_{sim} and dimension count N_{var} approximately linearly.

4 PERFORMANCE AND COMPUTING TIME ISSUES

The following section offers a performance comparison of the single-thread process implemented in JAVA language [20] and the massively parallelized solution using nVidia CUDA platform. The former was executed on the Intel Core i7-4700MQ processor. Its basic parameters are summarized by table 1:

	Core i7-4700MQ	
Architecture	Haswell	
Maximal frequency	3400	[MHz]
L1 cache per core	32	[KB]
L2 cache per core	256	[KB]
L3 cache share	6	[MB]
Bus bandwidth	25,6	[GB/s]
Computing performance	83	[GFLOPS]
Manufacturing technology	22	[nm]

Tab. 1: Parameters of the processor used.

The massively parallel solution on the nVidia CUDA platform was executed on three graphic cards: nVidia Quadro K1100M, GeForce GTX580 and GeForce GTX970. Their fundamental parameters are compared in table 2:

	K1100M	**GTX 580**	**GTX 970**	
Architecture	Kepler	Fermi	Maxwell	
Multiprocessor count	2	16	64	[-]
Shaders per MP	192	32	26	[-]
Shaders in total	384	512	1664	[-]
Core clock	716	770	1050	[MHz]
Memory clock (effective)	2800	4000	7012	[MHz]
Bus bandwidth	45	192	224	[GB/s]
DRAM capacity	2048	1536	4096	[MB]
Computing performance	542	1581	3494	[GFLOPS]
Manufacturing technology	28	40	28	[nm]

Tab. 2: Parameters of the graphic cards used.

For a basic comparison of computing performance of the mentioned hardware, one can consider the value of performance in FLOPS (**FL**oating point **OP**erations per **S**econd) which is a measure of computational power. Today's most powerful supercomputers reach up to tens of peta (10^{15}) FLOPS[4]. Conversely, the performance an ordinary desktop calculator varies around units of FLOPS.

Table 3 and figure 4 show the influence of point count N_{sim} on the execution time of 10^4 steps of dynamical simulation. In case of the single-thread JAVA process, roughly quadratic dependence of computational time t_{CPU} on the point count N_{sim} can be identified. The character of execution time of the parallelized solutions t_{K1100M}, t_{GTX580} and t_{GTX970} confirms the said assumption of linear dependency.

N_{sim}	N_{var}	t_{CPU} [s]	t_{K1100M} [s]	t_{GTX580} [s]	t_{GTX970} [s]
5	2	0.144	0.312	0.292	0.394
10	2	0.451	0.449	0.301	0.451
20	2	1.194	0.759	0.399	0.664
50	2	6.502	1.833	0.813	0.930
100	2	19.361	3.560	1.922	1.662
150	2	45.257	5.545	3.319	2.369
200	2	76.861	8.263	5.295	3.348

Tab. 3: Comparison of execution time dependency on point count N_{sim} ($N_{var} = 2$).

[4] The peak performance of world's most powerful supercomputer, Tianhe-2 (Guangzhou, China), is almost 34 PFLOPS.

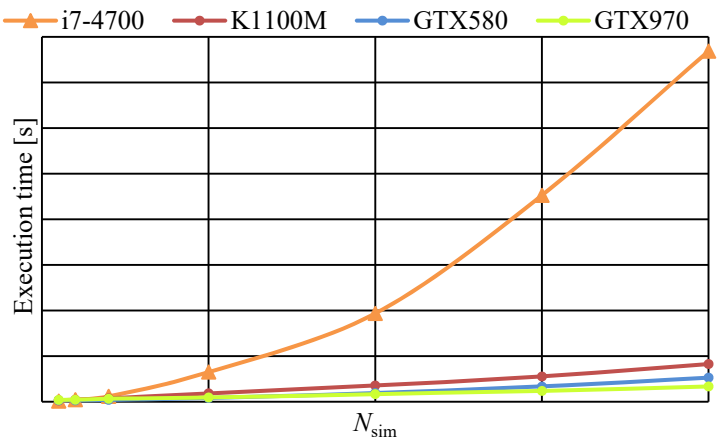

Fig. 4: The rise of execution time with point count N_{sim} ($N_{var} = 2$).

The influence of the dimension of design hypercube N_{var} on the execution time is illustrated by table 4 and figure 5. A steep growth in execution time with the dimension N_{var} while utilizing a single thread (JAVA) is evident.

N_{sim}	$N_{var,\ CPU}$	t_{CPU} [s]	$N_{var,\ CUDA}$	t_{K1100M} [s]	t_{GTX580} [s]	t_{GTX970} [s]
10	2	0.133	2	0.445	0.301	0.451
10	5	0.216	5	0.473	0.305	0.489
10	10	0.983	10	0.525	0.306	0.587
10	11	2.642	20	0.621	0.307	0.679
10	12	7.302	30	0.731	0.311	0.753
10	13	22.472	40	0.819	0.336	0.826
10	14	73.081	50	0.984	0.372	0.947

Tab. 4: Comparison of execution time dependency on hypercube dimension N_{var} ($N_{sim} = 10$).

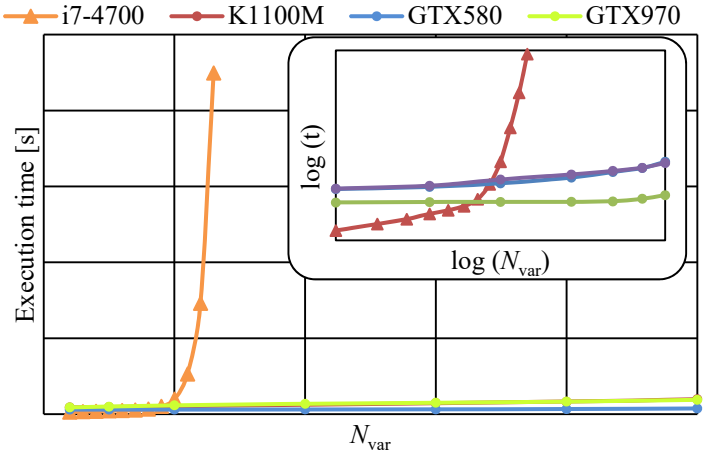

Fig. 5: The rise of execution time with the hypercube dimension N_{var} ($N_{sim} = 10$).

The steep growth in execution time for both N_{sim} and N_{var} effectively disqualifies the utilization of the single-thread JAVA implementation even for problems of fairly low complexity. The phenomenon of steep growth tends to appear approximately from the point count $N_{sim} = 10$ in dimension $N_{var} = 12$ and beyond. One can expect that this is associated with reaching the size of the allocated space in L1 cache of the CPU (32 KB per core). At that time, the process operates roughly with 1600 values of double-precision (binary64). That means the need of storing about 13 KB of data in the L1 cache. Since then, the process clearly begins to use the computer's operation memory which exhibits much longer access time.

On the other hand, the massively parallel solution seems to be highly suitable for solving such tasks for all maintained data is stored within the GPU's global DRAM memory (2048 / 1536 / 4096 MB) with rather large bandwidth. Further, let us consider that the global memory has the longest access time of all available memory types on the GPU. One can conclude that with a sufficient hardware equipment, the execution time of the parallel solution is extremely insensitive to increase in both the point count N_{sim} and the dimension of the hypercube N_{var}.

5 CONCLUSION

The paper deals with the approach of optimization of Design of Experiment (DoE) using a simulation of a discrete dynamical system of interacting particles which is analogical to the Audze-Eglājs criterion for optimization of the design. The implemented simulation of dynamical system is based specifically on the periodical modification of the AE criterion that already exhibited excellent results of point layout within the design space.

In the second part of the paper, two performed approaches of solution implementation were presented: the object oriented, single-thread process using the JAVA environment and the massively parallel solution based on the nVidia CUDA platform. For both, the sensitivity of the execution time on the point count N_{sim} and on the hypercube dimension N_{var} was studied.

It was shown that the approach of massive parallelization is highly suitable for solving of such tasks due to its immense computational power and low dependency on the complexity of solved task. Therefore, the current implementation in the CUDA C/C++ language will be further developed and used in the context of collective's research activities.

ACKNOWLEDGEMENT

The authors acknowledge financial support provided by the Czech Ministry of Education, Youth and Sports under project No. LO1408 "AdMaS UP - Advanced Materials, Structures and Technologies" under the "National Sustainability Programme I".

LITERATURE

[1] METROPOLIS, N.; ULAM, S. The Monte Carlo method. *Journal of the American statistical association*, 1949, 44.247: 335-341.

[2] CONOVER, W. M. On a better method for selecting input variables. *Helton JC, Davis FJ, editors*, 1975.

[3] IMAN, R. L.; DAVENPORT, J. M.; ZEIGLER, D. K. *Latin hypercube sampling (program user's guide). [LHC, in FORTRAN]*. Sandia Labs., Albuquerque, NM (USA), 1980.

[4] ELIÁŠ, J., VOŘECHOVSKÝ, M. Modification of the Audze-Eglājs criterion to achieve a uniform distribution of sampling points. *Advances in Engineering Software,2016,* 100: 82—96.

[5] BRANDOLINI, L., et al. On the Koksma–Hlawka inequality. *Journal of Complexity*, 2013, 29.2: 158-172.

[6] MCKAY, M.D., CONOVER, W.J., BECKMAN, R.J. *Technometrics,* 1979*,* 21, 239.

[7] IMAN, R. L.; CONOVER, W. J. Small sample sensitivity analysis techniques for computer models with an application to risk assessment. *Communications in statistics-theory and methods*, 1980, 9.17: 1749-1842.

[8] VOŘECHOVSKÝ, M.; NOVÁK, D. Correlation control in small-sample Monte Carlo type simulations I: A simulated annealing approach. *Probabilistic Engineering Mechanics*, 2009, 24.3: 452-462.

[9] BATES, S. J.; SIENZ, J.; LANGLEY, D. S. Formulation of the Audze–Eglais uniform Latin hypercube design of experiments. *Advances in Engineering Software*, 2003, 34.8: 493-506.

[10] TOROPOV, S.B., QUERIN, O. *Proceedings of Ninth International Conference on the Application of Artificial Intelligence to Civil, Structural and Environmental Engineering*, 2007, 1-12.

[11] AUDZE, P.; EGLAIS, V. New approach for planning out of experiments. *Problems of Dynamics and Strengths*, 1977, 35: 104-107.

[12] JOHNSON, M. E.; MOORE, L. M.; YLVISAKER, D. Minimax and maximin distance designs. *Journal of statistical planning and inference*, 1990, 26.2: 131-148.

[13] FANG, K. T.; WANG, Y. Number-Theoretic Methods in Statistics, *Chapman & Hall. New York*, 1994.

[14] FANG, K.; MA, Ch; WINKER, P.. Centered L_2-discrepancy of random sampling and Latin hypercube design, and construction of uniform designs. *Mathematics of Computation*, 2002, 71.237: 275-296.

[15] FANG, K.; MA, Ch. Wrap-around L 2-discrepancy of random sampling, Latin hypercube and uniform designs. *Journal of complexity*, 2001, 17.4: 608-624.

[16] ROMERO, V.J., BURKHARDT, J.V., GUNZBURGER, M.D., PETERSON J.S. Reliability Engineering and System Safety, *The Fourth International Conference on Sensitivity Analysis of Model Output*, 2006, 91: 10-11.

[17] ŠMÍDOVÁ, M. Performance comparison of methods for design of experiments for analysis of tasks involving random variables. *Brno University of Technology*, 2014

[18] VOŘECHOVSKÝ, M. Hierarchical refinement of Latin hypercube samples. *Computer-Aided Civil and Infrastructure Engineering*, 2015, 30.5: 394-411.

[19] VOŘECHOVSKÝ, M.; ELIÁŠ, J. Improved formulation of Audze-Eglājs criterion for space-filling designs. *12th International Conference on Applications of Statistics and Probability in Civil Engineering*, 2015.

[20] FRANTÍK, P., VOŘECHOVSKÝ, M. Modification of the FyDiK n-D application for dynamical simulations of discrete dynamical particle systems in JAVA environment.

[21] CHE, S., et al. A performance study of general-purpose applications on graphics processors using CUDA. *Journal of parallel and distributed computing*, 2008, 68.10: 1370-1380.

[22] BAYOUMI, A., et al. Scientific and engineering computing using ATI stream technology. *Computing in Science & Engineering*, 2009, 11.6: 92-97.

[23] FRANTÍK, P.; VESELÝ, V.; KERŠNER, Z. Parallelization of lattice modelling for estimation of fracture process zone extent in cementitious composites. Advances in Engineering Software, 2013, 60: 48-57.